|  |  |  |
| --- | --- | --- |
| Existing Problem | Existing solutions | Open Challenges |
| - Manual collection and labeling of data are inefficient and insufficient | - Surprise Adequacy for Deep Learning Systems (SADL) measures the surprise of inputs compared to training data to guide test input selection. | - Fine-grained test coverage criteria are needed to guide the selection of individual inputs for improving DL system accuracy​​​​. |
| . - Existing coverage criteria based on neuron activation values are not fine-grained enough to capture subtle DL behaviors. | - Likelihood-based Surprise Adequacy (LSA) and Distance-based Surprise Adequacy (DSA) are introduced to quantify input novelty and behavior difference | - Developing scalable, efficient, and generalizable end-to-end testing methods that can handle the complexity and safety-critical nature of DNN applications​​. |
| - Evaluations focus more on the correlation between adversarial examples and criteria rather than on actual DL system testing​​. | ​​. - Introduction of neuron coverage and other criteria to fill gaps in DNN testing methods​​. | - Need for criteria that are applicable to various DNN architectures and capable of identifying more nuanced behaviors​​. |
| - Traditional software testing criteria like code coverage cannot be applied to DNNs due to their lack of explicit control-flow structure​​. | - Proposing coverage criteria that account for inter-layer and intra-layer neuron relationships, and a 2-way coverage criterion for more comprehensive logic testing​​. | - Enhancing the efficiency and relevance of test input generation methods, and establishing reliable oracles for correctness testing​​. |
| - Existing testing methods are either too coarse or fail to account for all possible DNN behaviors​​. |  | systematic testing coverage criteria that can detect defects and evaluate test quality effectively in DNNs​​. |
| traditional white-box testing techniques that aim to increase structural coverage [4] is not very useful for DL systems, as their behaviour is not explicitly encoded in their control flow structures. |  | Automated and systematic testing of DL systems with thousands of neurons and millions of parameters for all corner cases is extremely challenging​​. |

Coverage Criteria : Traditional software testing methods fail when applied to DNNs because the code for deep neural networks holds no information about the internal decision-making logic of a DNN , unlike traditional software, DNNs do not have a clear control-flow structure. They learn their decision policy through training on a large dataset, adjusting parameters gradually using several methods to achieve desired accuracy. Consequently, traditional software testing methods like functional coverage, branch coverage, etc. cannot be applied to DNNs, thereby challenging their use for safety- critical applications.

|  |  |  |
| --- | --- | --- |
| Existing Coverage Methods | Description | Limitation |
| Neuron Coverage | Measures the model's logic use by counting activated neurons from test inputs. | It doesn't capture all potential DNN behaviors and can achieve high coverage with few inputs, it is a coarse measure not sufficient for thorough DNN evaluation. |
| k-Multisection Neuron Coverage | Divides neuron activation values observed during training into k buckets and counts how many buckets are covered by a set of inputs. | Loses information on activations beyond the observed range during aggregation, making it hard to evaluate the relative value of each input for enhancing DL system accuracy. |
| DeepCover | Considers condition-decision dependencies between adjacent DNN layers. | Its applicability is limited to small, feedforward, fully-connected networks and doesn't generalize to complex architectures like RNNs or LSTMs. |
| DeepCT | Inspired by combinatorial testing, it assesses logic use by the fraction of neurons activated in each layer. | Lacks consideration for inter-layer relationships and hasn't been proven to scale to real-world DNNs with varied layer types, indicating potential limitations in broader usage. |

1. Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. 2017.
2. Lei Ma, Felix Juefei-Xu, Jiyuan Sun, Chunyang Chen, Ting Su, Fuyuan Zhang, Minhui Xue, Bo Li, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. Deepgauge: Comprehensive and multi-granularity testing criteria for gauging the robustness of deep learning systems. *CoRR*, abs/1803.07519, 2018.
3. Youcheng Sun, Xiaowei Huang, and Daniel Kroening. Testing Deep Neural Networks. 2018.
4. Lei Ma, Fuyuan Zhang, Minhui Xue, Bo Li, Yang Liu, Jianjun Zhao, and Yadong Wang. Combinatorial Testing for Deep Learning Systems. 2018.

Best input generation: Generating or selecting test inputs in a guided manner usually has two major goals - maximizing the number of un- covered faults, and maximizing the coverag

|  |  |  |
| --- | --- | --- |
| Existing Methods | Description | Limitation |
| Joint Optimization | Modifies an existing input through image manipulations recursively until it triggers different behavior in the model. [1] | Time-consuming and produces a low ratio of impactful test inputs compared to the total tested/generated. |
| Greedy Search | Applies random transformations to an existing test input until a suitable test input is identified. [2] | Similar to joint optimization, it is also time-intensive and results in a low number of effective test inputs relative to the total tested. |

1. Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. 2017.
2. Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars. 2017.

Best Oracle

|  |  |
| --- | --- |
| Existing Approaches | Limitations |
| collecting as much real-world data as possible and manually labeling it to check for correctness. | process requires a lot of manual effort. |
| compares outputs across multiple DNNs for the same task, identifying discrepancies as corner cases [2]. | this can misclassify inputs if all models agree, due to shared biases or errors. This comparative approach is further limited to tasks with multiple reliable models, which may not always be available, especially in innovative or specialized applications. |

1. Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. 2017.